Electronic Preprint for Journal of Information Processing Vol.20 No.2

Regular Paper

An Adaptive Route Selection Mechanism Per Connection Based on Multipath DNS Round Trip Time on Multihomed Networks

Yong Jin[1] Nariyoshi Yamai[2,a)] Kiyohiko Okayama[2] Motonori Nakamura[3]

Received: June 26, 2011, Accepted: December 16, 2011

Abstract: With the explosive expansion of the Internet, many fundamental and popular Internet services such as WWW and e-mail are becoming more and more important and are indispensable for the human's social activities. As one technique to operate the systems reliably and efficiently, the way of introducing multihomed networks attracts much attention. However, conventional route selection mechanisms on multihomed networks reveal problems in terms of properness of route selection and dynamic traffic balancing which are two key criteria of applying multihomed networks. In this paper, we propose an improved dynamic route selection mechanism based on multipath DNS (Domain Name System) round trip time to address the existing problems. The evaluation results on the WWW system and the e-mail system indicate that the proposal is effective for a proper route selection based on the network status as well as for dynamic traffic balancing on multihomed networks and we also confirmed the resolution of problems that occur in the case of conventional mechanisms.

Keywords: multihomed network, routing, DNS, load balancing, fault-tolerance

1. Introduction

With the rapid progress and popularity of the Internet, many Internet services such as the WWW and the e-mail have become fundamental elements of the cyberspace and are apparently indispensable for human's social activities. Especially nowadays, whatever we can imagine is being propagated through the WWW and many kinds of important information and messages attached with bulk files are being exchanged via the e-mail frequently. Obviously it is extremely necessary to run the Internet systems reliably and efficiently. From this viewpoint, if the systems are constructed in a single-homed network which has only one single physical link (backbone) to the Internet, they are vulnerable to the break of connection to the Internet. Accordingly, as a solution to provide more reliable and efficient Internet services, multihomed networks which have multiple physical links to the Internetattract much attention. By using multihomed networks, the Internet services can be provided via multiple different routes so that it is possible to improve the fault-tolerance of the systems based on redundant links to the Internet and to improve the system performance by traffic balancing among multiple backbones.

In this research, we focus on the practical use of multihomed network for route selection [*1] of incoming connections (initiated from the client side to the multihomed network side) which are more difficult than that in outgoing connections (initiated from the multihomed network side to the client side). That is, for incoming connections, when we deploy some route selection policies in the multihomed network side, it is difficult to make the client side select the proper route based on the policies. On the other hand, for outgoing connections, it is comparatively easy to control the route selection on the multihomed network side by introducing appropriate schemes, for example, by means of the method we have proposed [1]. Furthermore, for incoming connections, it is also possible to introduce new schemes into the client side to make the client select the proper route based on the policies of the multihomed network side. However this approach may cause a new problem in terms of effectivity because we can hardly customize all client sides configuration.

A multihomed network offers several advantages: multiple links are effective for improving the fault-tolerance and multiple links to different ISPs improve the network performance, etc. Note that it is not desirable that the end user performs the route selection based on the knowledge of multiple links. Thus, so far there have been several transparent techniques developed to construct a multihomed network, which are, the way of obtaining an AS (Autonomous System) number [2] (Technique 1), the way of using the NAT (Network Address Translation) [1] (Technique 2) and the way of using the ALG (Application Level Gateway) [3] (Technique 3). Basically, all of the above techniques are applicable to the topic we discuss in this paper and each of them has

[1] Graduate School of Natural Science and Technology, Okayama University, Okayama 700–8530, Japan
[2] Center for Information Technology and Management, Okayama University, Okayama 700–8530, Japan
[3] National Institute of Informatics, Chiyoda, Tokyo 101–8430, Japan
[a)] yamai@cc.okayama-u.ac.jp

[*1] This is called route selection because it is independent of the hop-by-hop routing protocols and just selects one proper route, such as an ISP (Internet Service Provider), by selecting one of the ALGs which are specified with different IP addresses.

its own characteristics. For Technique 1, it is only applicable for an outgoing connection since it is strictlyeffected by the BGP policies deployed on the client side. Likewise, Technique 2 is basically an outgoing connection oriented technique and is inapplicable for an incoming connection, either, unless it does not use the NAPT (Network Address Port Translation). For Technique 3, generally it may require the ALGs for every application. These techniques only construct a multihomed network but an intelligent route selection mechanism is required specifically.

Accordingly, it is barely possible to improve the reliability and the efficiency of the Internet systems just by constructing a multihomed network, in the meantime, it is required to perform a proper route selection. Basically, it is desirable that the break down link can be avoided automatically during communications between the internal network and the external networks and the network traffic can be balanced based on the backbone usage rate as well. Furthermore, even in a low load situation, it is also necessary to use the backbone which has a wide bandwidth (called fast) or the one with a low latency (called near) to reduce the transmission delay, which eventually contributes to cut down the traffic of the whole cyberspace. Based on the above consideration, we call this kind of route a proper route in the rest of this paper.

As mentioned above, the conventional multihoming techniques are not applicable for the route selection of incoming connections or some customizations are required on the client side which also have an effectivity problem. Thus in this research, we focus on a common criterion "transmission delay" and purpose to get a solution for the route selection of an incoming connection, which is not only unrestricted to particular applications but also there is no need to customize the client side. Considering the low cost of the maintenance and the implementation, we choose the technique of using the ALG for multihoming a network.

To date there have been many approaches developed to accomplish the above goals. One of the well known representative techniques is the DNS Round Robin (DNS RR for simplicity) [4]. The DNS RR was originated for the purpose of load balancing among multiple equivalent replica servers and it is also applicable to route selection on multihomed networks by setting an ALG at the junction of each backbone. However, the DNS RR has some drawbacks in terms of the properness of route selection and dynamic traffic balancing.

On the other hand, we have proposed an improved method named DNS Response Multiplication (DRM) [5] to solve the problems. In DRM, multiple same kind DNS resource records are replied to every single query via multiple routes and the one that first arrives at the queried side will be valid. Consequently, the backbone through which the fastest DNS response was delivered will be used by the users. However, DRM also has a problem in terms that it is not applicable to the networks with ingress filtering [6] [*2]. Moreover, DRM works based on the network condition in the outbound direction which turns out as a problem in the e-mail system for the inbound e-mail delivery.

Thus in this paper, we propose a new dynamic route selection mechanism to address the above problems. Basically, the proposal is adaptable to ingress filtering and works based on the network condition in a proper direction according to the application types. For example, the proposal works differently for the outbound type application whose main transmission is in the outbound direction such as the WWW and the inbound type application whose main transmission is in the inbound direction such as the inbound e-mail delivery in the e-mail system.

In this mechanism, we focus on the domain name resolution which is performed right before the users use the Internet services. During the domain name resolution, we inspect the condition of each route in a proper direction by measuring the latency using DNS queries as well as the corresponding responses, then let the users use the most proper route. With this mechanism, it is possible not only to take advantages of the multihomed network but also to solve the existing problems in conventional route selection mechanisms.

2. Route Selection on Multihomed Networks and the Problems

2.1 Redundant Route Selection on Multihomed Networks

A multihomed network is a kind of network that has multiple physical links to the Internet like the internal network shown in **Fig. 1**. In general, we reasonably consider that the systems constructed in a multihomed network possibly have better fault-tolerance and better performance than those constructed in a single-homed network. However, to achieve the above advantages, it is necessary to perform a proper route selection for the main transmissions of the Internet systems as mentioned in the previous section.

So far there have been two techniques well used for a route selection on multihomed networks, of which, one is the BGP (Border Gateway Protocol) [7] and the other is the way of using ALG.

The BGP is one of the most widely used routing protocols for a hop-by-hop routing mechanism and it is also applicable to the route selection on multihomed networks. In BGP, the routing operations or route selection procedures are handled completely by the IP layer so that there is no boring customization needed on specific applications such as the WWW system or the e-mail system. However, when we use the BGP to control a route selection, it cannot provide route information to every client but only to the nearest AS, and eventually the nearest AS decides the proper route based on its routing policies. Accordingly, the BGP is not applicable to adaptive route selection even though it is able to distribute network traffics to multiple links. Moreover, it needs to cooperate with other organizations to share the route information using the BGP so that a high level management skill is required

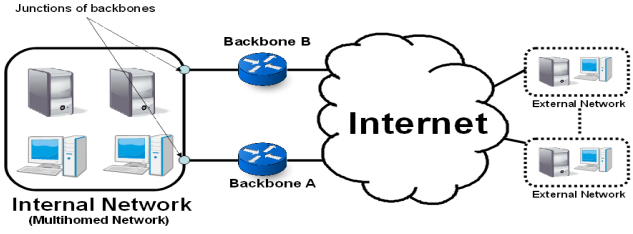

Fig. 1 Multihomed network configuration.

[*2] Ingress filtering is a technique used to make sure that incoming packets are actually from the networks that they claim to be from. Currently many ISPs are using this technique and the amount are getting increased.

and its maintenance cost of the BGP is extremely expensive as well.

On the other hand, in the way of using the ALG, the BGP may also be used as a routing protocol but it is not used for route selection. In this mechanism, an ALG is set at the junction of each backbone with a different IP address allocated and all traffics between the internal network and the Internet are transmitted via the ALGs. In other words, a different ALG uses a different route (such as a different ISP) for the communication with the Internet. Thus the route selection can be performed by letting the users select different ALGs no matter where the client is located. Obviously, although some customization needs to be added on the application layer, the way of using the ALG is relatively easy to realize and comparatively easy to maintain and operate against the BGP. Therefore, in the rest of this paper, we focus on the route selection mechanisms using the ALG.

2.2 Conventional Route Selection Mechanisms on Multihomed Networks and the Existing Problems

Within multihomed networks involving the ALG, a route selection is equivalent to an ALG selection since each ALG has a single global IP address allocated by each backbone and set at each junction. So far there have been several solutions proposed for load balancing in such an environment and they are also applicable to a route selection. One of the representative solution is DNS RR. In addition, DRM is an improved method we proposed to solve some problems in DNS RR.

2.2.1 DNS Round Robin

DNS RR is a technique of load balancing by provisioning multiple same kind DNS resource records on a single host name or domain name and replying them as a list in one response, in which the sequence is permuted each time to each query. This technique is applicable to a WWW system consisting of multiple replica servers as well as to an e-mail system consisting of multiple MXes (Mail eXchange) with the identical preference value. In these systems, there is no standard for deciding which one to be used, but in most implementations the first A record or MX record is used thus each server can be selected evenly. This technique is implemented in most DNS server programs such as BIND (Berkeley Internet Name Domain) [8], thus is widely available.

However, this technique has a drawback in terms of properness of route selection and dynamic traffic balancing. That is, it is possible to balance the load only evenly among multiple replica servers or balance the traffic averagely among backbones but it is impossible to make clients select the fastest or nearest backbone since the positions of clients and the network condition are not considered.

2.2.2 DistributedDirector

DistributedDirector (DD) [9] is one of the commercial products of Cisco Corporation and it has been developed for load balancing between multiple replica servers being located at different places. DD is capable of letting clients access the nearest server dynamically and transparently without customizing either the servers or the clients. Specifically, DD works as a DNS server and collaborates with the routers near to each replica server and retrieves the route information of the replica serer nearest to the client, and finally replies it as a DNS response. DD uses the AS path length (BGP) and metrics of Cisco's specific routing control protocols such as IGRP (Interior Gatweay Routing Protocol) and EIGRP (Enhanced IGRP) [*3] to judge the distance between the replica server and the client. In a multihomed environment we can use DD to perform the route selection based on the bandwidth and the network conditions.

However, when the AS path length is used as the criterion of the route selection, DD is not able to make the client select the proper route based on the bandwidth or the network conditions dynamically since BGP does not consider the usage rate of the bandwidth and the congestion condition. Also, as we mentioned in Section 2.1, it requires an incredibly high cost to introduce, implement and maintain BGP. Furthermore, the IGRP or EIGRP is Cisco's proprietary protocol, which means the routing controls between all the clients and replica servers have to support the specific protocols, which is not realistic in real network environments. As a result, we consider DD does not satisfy our purpose.

2.2.3 TENBIN

TENBIN [10] is another route selection solution for load balancing between multiple replica servers similar to DD. TENBIN also works as a DNS server, but it uses an external program called RADIX [10] which works based on BGP to collect route information from routers near to each replica server. Likewise, TENBIN finally replies the nearest replica server to the client as a DNS response. TENBIN uses the AS path length as the metric and it also can use multiple route selection policies such that some servers are restricted for access of domestic clients, but others for overseas clients. It makes TENBIN control the criteria of server selection more flexibly. Accordingly, TENBIN can also be used for route selection on multihomed network.

However, as mentioned, TENBIN uses only the AS path length as the metric. Thus although load balancing can be performed in some way on a multihomed network, TENBIN cannot select the proper route based on the bandwidth or network conditions dynamically. Moreover, since TENBIN uses BGP, the applicable scale is restricted to the area supporting BGP and the cost issue of introduction, maintain and administration arises, too. Finally, when TENBIN exploits its route selection policies, they have to be extended to every client side to obtain the route information which is impossible in reality. With the above reasons, obviously TENBIN does not fit for the respect of our approach.

2.2.4 DNS Response Multiplication

DRM is a dynamic route selection mechanism for multihomed networks, we have proposed to address the existing problems in DNS Round Robin. In this mechanism, a Query Duplicator is set for receiving DNS queries from the Internet. Also an ALG and a DNS server are set at the junction of each backbone and multiple same type DNS records with different contents are configured on each DNS server. Basically, each set of an ALG and a DNS server uses a different route to communicate with the Internet. In other words, each DNS server claims that the ALG at the same junction with itself has a high priority, and then replies the DNS

[*3] Bandwidth, delay, payload, reliability and MTU (Maximum Transmission Unit) are referenced in the metric.

response simultaneously to every single DNS query.

In the DNS scenario, one DNS response matches one DNS query with its query ID, which means, when there exist multiple DNS responses corresponding to one single DNS query, the one that first arrives at the querying side becomes valid and the others get discarded. Accordingly, the backbone through which the valid DNS response (the first arriving one) passed will get used by the users for the main transmission of the Internet services (for example, the WWW propagation or the e-mail delivery). Thus, the clients can use the proper route according to the network condition and traffic balancing is possible as well.

However, the DRM has a problem in the sense that it is not applicable to the networks with ingress filtering. In DRM, the DNS queries are received by the Query Duplicator then the Query Duplicator duplicates and transfers the DNS queries to each DNS server and DNS responses are replied by each DNS server finally. Consequently, some DNS response packets can be reasonably filtered (normally discarded) by ingress filtering since the DNS response packets are replied via multiple routes using the same source IP address. Additionally, the DRM also has a problem that it is mainly sensitive to the network condition in the outbound direction so that it is not much effective for inbound type applications such as the inbound delivery in the e-mail system.

3. Adaptive Route Selection Mechanism per Connection Based on Multipath DNS Round Trip Time

In this section we propose an adaptive route selection mechanism by measuring the multipath DNS Round Trip Time (RTT) to solve the problems in conventional methods. In the proposal mechanism, we set an internal DNS server (authoritative DNS server) at the junction of each backbone and add external features to them in order to make an external DNS server (recursive and caching DNS server) query multiple times during the domain name resolution. Through the multiple query process, we can finally measure the outbound or inbound latencies of backbones using the DNS query and response packets and let the client select the most proper route. We discuss the details in the following.

3.1 Latency Measurement Using DNS

To solve the ingress filtering problem, we take away the Query Duplicator and use the DNS servers only to inspect the network condition by measuring the multipath DNS round trip time in our proposal mechanism. In fact, the ideal solution is to measure the latency between each ALG and the client before the main transmission and let the client select the most proper one. However, this solution needs collaboration with the client side which means the customization on the client side is unavoidable. Therefore, in the proposal mechanism, we consider an indirect way to measure the latency of each route between the server and the client. Next, we present the detail scenario in the outbound direction and the inbound direction using **Fig. 2** and **Fig. 3**, respectively.

For the simplicity we illustrate only the DNS servers here. For the latency measurement in the outbound direction we consider to measure two RTTs. As shown in Fig. 2, RTT1 is the RTT starting from the I_DNS1, via the E_DNS and returning back to the I_DNS2. Similarly, RTT2 is the RTT starting from the I_DNS2, via the E_DNS and returning back to the I_DNS2. Here, we can assume the latency from the E_DNS to the I_DNS2 (step 3 and step 5) is the identical portion in RTT1 and RTT2. Accordingly, we can compare the latency from the I_DNS1 to the E_DNS (step 2) and the latency from the I_DNS2 to the E_DNS (step 4) by comparing RTT1 and RTT2. Finally, we can decide the route with a low latency in the outbound direction.

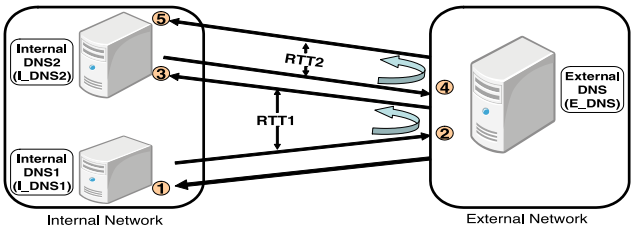

Fig. 2 Outbound latency measurement.

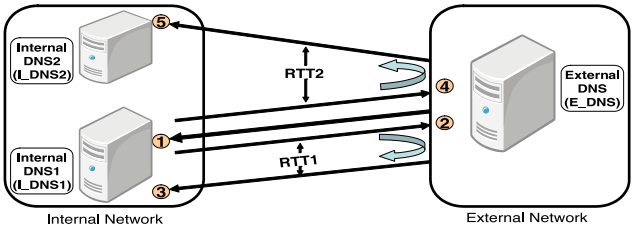

Fig. 3 Inbound latency measurement.

Similarly, we can apply this approach for the latency measurement in the inbound direction on a multihomed network with a little customization. For the latency measurement in the inbound direction, we also consider to measure two RTTs. As shown in Fig. 3, RTT1 is the RTT starting from the I_DNS1, via the E_DNS and returning back to the I_DNS1, and RTT2 is the RTT starting from the I_DNS1, via the E_DNS and returning back to the I_DNS2. Here, we can assume the latency from the I_DNS1 to the E_DNS (step 2 and step 4) is the identical portion in RTT1 and RTT2 so that we can compare the latency from the E_DNS to the I_DNS1 (step 3) and the latency from the E_DNS to the I_DNS2 (step 5) by comparing RTT1 and RTT2. Finally, by this comparison we can decide the route with a low latency in the inbound direction.

As we described above, in the proposal mechanism, we need to make the external DNS server query multiple times to measure the TTLs without customizing the external DNS server. In the DNS protocol, the CNAME (Canonical NAME) record has the characteristic that makes the target DNS server query again. Thus in the proposal mechanism, we use CNAME records as temporary replies for one domain resolution. During the domain name resolution, we add the timestamps such as the reply time and the RTT1 as a part of a CNAME record when the internal DNS servers return temporary replies. Accordingly, when the external DNS server queries the same type of DNS record of the CNAME, the internal DNS servers can calculate the RTTs and compare the RTTs as well by parsing the CNAME record.

Note that both of the RTT1 measurement in Fig. 2 and the RTT2 measurement in Fig. 3 involve two different physical hosts (I_DNS1 and I_DNS2) so that in order to measure the RTT1 and

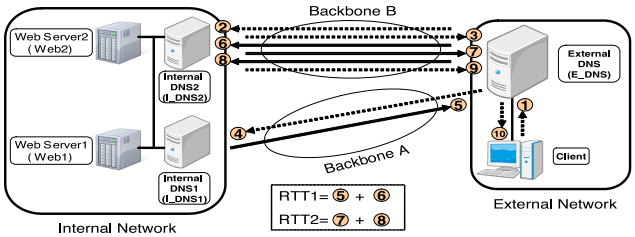

Fig. 4 Latency measurement in the outbound direction.

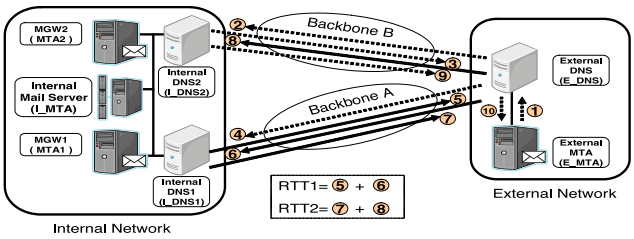

Fig. 5 Latency measurement in the inbound direction.

the RTT2 precisely they need to synchronize their system timer. Fortunately, we can easily address the synchronization using NTP (Network Time Protocol) [11].

3.2 Latency Measurement in the Outbound Direction

We pick up the WWW system which mainly have outbound traffic transmissions and present the procedures of the proposal using **Fig. 4**. In this example, we assume the I_DNS1 in the Internal Network has the authority of zone "www.example.com" and the I_DNS2 has the authority of zone "example.com." We show the detailed steps of the initial procedure in the following (The step numbers correspond to the numbers illustrated in Fig. 4).

(1) The Client queries the A record of "www.example.com" to the E_DNS.
(2) E_DNS queries the A record to I_DNS2 (authoritative DNS server).
(3) I_DNS2 replies I_DNS1 as NS record of the domain "www.example.com."
(4) E_DNS queries the A record of "www.example.com" to I_DNS1.
(5) I_DNS1 replies "*timestamp1*.example.com" as the CNAME record where *timespamp1* is the reply time of this response.
(6) E_DNS queries the A record of "*timestamp1*.example.com" to I_DNS2 since it has the authority of zone "example.com."
(7) I_DNS2 replies "*timestamp2*.*rtt1*.example.com" as the CNAME record where *timestamp2* is the reply time of this response and *rtt1* is the RTT1 described in Section 3.1 (see Fig. 2).
(8) E_DNS queries the A record of "*timestamp2*.*rtt1*.example.com" to I_DNS2.
(9) The I_DNS2 calculates RTT2 described in Section 3.1 (see Fig. 2) using the *timestamp2* and replies the server of the smaller RTT.
(10) The E_DNS replies the response to the Client.

With the above procedures, the internal web system can let the Client access the Web Server via the route with a low latency in the outbound direction. In the initial operation, the NS records of the domains "www.example.com" and "example.com" can be cached in the E_DNS so that the E_DNS can query the A record of the domain "www.example.com" to the I_DNS1 directly from the second try. Furthermore, the TTL of the A record of the domain "www.example.com" as well as those of the temporary domains such as "*timestamp1*.example.com" and "*timestamp2*.*rtt1*.example.com" are set to 0 [*4] to avoid the records being cached in the E_DNS. Accordingly the Client queries the A record of the domain "www.example.com" every time of accessing to the Internal Web server.

[*4] If there are some troubles with 0 as the TTL value, another small value like 1 or 2 still works well.

3.3 Latency Measurement in the Inbound Direction

As another example, we use the inbound e-mail delivery in which the main traffic transmission is in the inbound direction to present the whole procedures of the proposal using **Fig. 5**. Similarly, in this example we assume the I_DNS1 in the Internal Network has the authority of zone "mail.example.com" and the I_DNS2 has the authority of zone "example.com." We show the detail steps of the initial procedure in the following (The step numbers are corresponding to numbers illustrated in Fig. 5). Since the first four steps are similar to those in the WWW system (just change the query for the A record into that for the MX record of "mail.example.com"), thus we omit the part here.

(5) I_DNS1 replies the domain name "*timestamp1*.mail.example.com" as the CNAME record where *timestamp1* is the reply time of this response.
(6) E_DNS queries the MX record of the domain name "*timestamp1*.mail.example.com" to I_DNS1 since it has the authority of zone "mail.example.com."
(7) I_DNS1 replies "*timestamp2*.*rtt1*.example.com" as the CNAME record again. The *timestamp2* is the reply time of this response and the *rtt1* is RTT1 described in Section 3.1 (see Fig. 3).
(8) E_DNS queries the MX record of the domain "*timestamp2*.*rtt1*.example.com" to I_DNS2 since it has the authority of zone "example.com."
(9) I_DNS2 calculates the *rtt2* which means the RTT2 described in Section 3.1 (see Fig. 3) using the *timestamp2* and replies the MTA of the smaller RTT.
(10) E_DNS replies the response to E_MTA.

With these procedures, the internal e-mail system can receive the inbound e-mail via the route with a low latency in the inbound direction. During the initial operation, the NS records of the domains "mail.example.com" and "example.com" can be cached in the E_DNS so that the E_DNS can query the MX record of the domain "mail.example.com" to the I_DNS1 directly from the second try. Likewise, we also set the TTLs of temporary domains to 0 and accordingly the Client queries the MX record of the domain "mail.example.com" every time of sending e-mail to the Internal Mail Server.

3.4 Fault-tolerance

Since a better fault-tolerance is one of the important advantages of multihomed networks, we also consider the feature in the proposal mechanism. We have designed the fault-tolerance

feature based on the practical use of the DNS software, thus during the process of a domain resolution, the proposal mechanism is able to predict the backbone fault and let the client avoid using the unavailable route for main transmission transparently by virtue of fallback operation with multiple NS records. This section describes the details of fault-tolerance.

Figure 6 shows the network configuration of a prototype system with the fault-tolerance feature integrated. To take full advantage of redundant backbones of multihomed network, we additionally implement one backup DNS server at each edge of the backbone. Each backup DNS server is the backup of each main DNS server which is set at the edge of the other side backbone as shown in the figure. In normal cases (both routes are available), only the main DNS serves work domain resolutions as well as RTT measurements without involving the backup DNS servers. The backup DNS servers only work when one of the routes becomes unavailable and do not measure the RTTs but just reply the available ALG as a DNS response directly when they receive DNS queries. For example, when I_DNS1_backup receives a DNS query for the MX record of the domain "mail.example.com" or the A record of the domain "www.example.com," it means that some problem maybe occurs on the route of Router1 by some reason that the destination server will be redirected to MTA2/Web2 which is connected to the route of Router2 by the I_DNS1_backup, and so does the I_DNS2_backup.

Here comes up a problem: how to control the system that the main DNSes serve in normal cases and the backup DNSes serve only when troubles happen on one of the links. In some DNS server softwares such as BIND and UNBOUND, a metric called *roundtrip time* is used to choose one NS record when there exist multiple authoritative NS records for the same domain. Here, the *roundtrip time* is a measurement of how long a remote name server takes to respond to queries. Each time a BIND name server sends a query to a remote name server, it measures the *roundtrip time* for it. When a recursive DNS server has to choose one among multiple NS records to query to, it simply chooses the one with the lowest *roundtrip time*. Based on this feature, we can adjust the *roundtrip time* against the DNS servers at the server side (internal network) and eventually the *roundtrip times* against main DNS servers are almost always less than those of backup DNS servers. Consequently, the main DNS servers almost always get used in normal cases, and when one of the routes breaks, the backup DNS server on the other side gets used since the external DNS server cannot get any response from the main DNS server at the broken route.

So far as we verified, this scenario is not applicable to some implementations other than BIND and UNBOUND. Although the fault-tolerance feature is restricted to BIND and UNBOUND, the proposal mechanism still works as well as DNS round robin even with other implementations.

4. Implementation and Evaluation

We implemented a prototype system of the proposal mechanism and picked up two most fundamental and popular Internet services, WWW and e-mail, as the representatives of outbound type applications and inbound type applications, respectively, and evaluated the features of a proper route selection and a dynamic traffic balancing on the multihomed network.

4.1 Implementation

The configuration of the prototype system is shown in **Fig. 7**. We set two internal servers each of which runs one internal DNS server (I_DNS1 and I_DNS2) and MGW (MTA1 and MTA2) as well as a web server (Web1 and Web2), respectively in the Internal Network. Each server is configured to communicate with the External Network via a different route, as shown in the configuration, through Router1 and Router2, respectively. We also set three FTP servers to generate the network traffic during evaluations.

The I_DNS1 and I_DNS2 are the two Internal main DNS servers described in Sections 3.2 and 3.3. We implemented the I_DNS1 and the I_DNS2 using BIND and a customized Perl DNS server module (Net::DNSServer::Proxy, PerlDNS for simplicity) [12]. The new feature in the combined DNS system is that it can reply a different CNAME record against the same domain name each time it receives a DNS query. Specifically, it can reply a CNAME record which includes the time when the response is replied in millisecond as a label. We set BIND as a normal DNS server and PerlDNS as a DNS forwarder. Normally, the PerlDNS listens and forwards DNS queries to the BIND and replies DNS responses from the BIND. For some specific queries such as the MX record query for an e-mail server or the A record query for a WWW server, the PerlDNS customizes the response packet before replying it to the external DNS server.

4.1.1 Zone Configuration

The following describes zone configurations and operation steps of the prototype system on a WWW system and an e-mail system respectively.

- RootDNS
  - Set two NS records for domain "example.com" on RootDNS

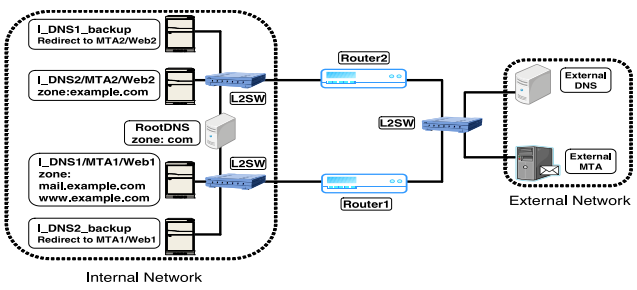

Fig. 6 Network configuration for fault-tolerance.

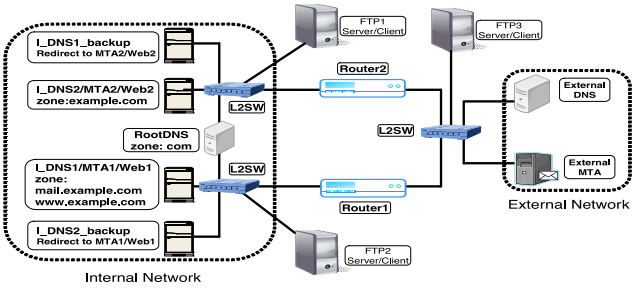

Fig. 7 Experimental Network.

example.com 86400 IN NS I_DNS2
86400 IN NS I_DNS2_backup

– Set two NS records for domain "mail.example.com" and domain "www.example.com" on the I_DNS2

mail.example.com 86400 IN NS I_DNS1
86400 IN NS I_DNS1_backup
www.example.com 86400 IN NS I_DNS1
86400 IN NS I_DNS1_backup

- Latency measurement in I_DNS1
  - For a WWW system
    (1) Reply "*timestamp1*.example.com" (see Section 3.2) as a CNAME record for A record query of the domain "www.example.com," where *timestamp1* describes the time when this response is replied.
  - For an e-mail system
    (1) Reply "*timestamp1*.mail.example.com" (see Section 3.3) as a CNAME record for the MX record query of the domain "mail.example.com," where *timestamp1* describes the time when this response is replied.
    (2) Reply "*timestamp2.rtt1*.example.com" (see Section 3.3) as a CNAME record for the MX record query of the domain "*timestamp1*.mail.example.com," where *timestamp2* describes the time when this response is replied and *rtt1* describes the RTT1 measured.
  - Works as normal DNS for other queries
- Response decision in I_DNS2
  - For a WWW system
    (1) Reply "*timestamp2.rtt1*.example.com" (see Section 3.2) as a CNAME record for the A record query of the domain "*timestamp1*.example.com," where *timestamp2* means the time when this response is replied.
    (2) Calculates RTT2, then decides the A record by comparing RTT1 and RTT2, finally replies the response.
  - For an e-mail system
    (1) Calculates RTT2, then decides the MX record by comparing RTT1 and RTT2, finally replies the response.
  - Works as normal DNS for other queries

Based on above operations, the prototype system works as we expected in the proposal mechanism. Note that in this system, all internal DNS servers are stateless. In other words, all information that is required to reply a DNS response is embedded in the DNS query packet so that the internal DNS servers are not needed to wait for the status of the domain name resolution. Accordingly, even the internal DNS servers receive multiple DNS queries at the same time, they can process these queries independently.

4.1.2 Delay Setting

We set a delay time on the DNS responses replying from the backup DNS servers in order to increase the *roundtrip time*. For simplicity of implementation, we set the delay time using a dummynet [13] on each backup DNS server. Specifically, we set the delay time on the DNS response packets sent out from the backup DNS servers. We performed some preliminary experiments by trying 100 domain name resolutions (MX record queries for domain "mail.example.com" and A record queries for "www.example.com") to decide the best delay time to add. **Figure 8** shows the relationship of DNS server working status versus delay time set on the backup DNS servers.

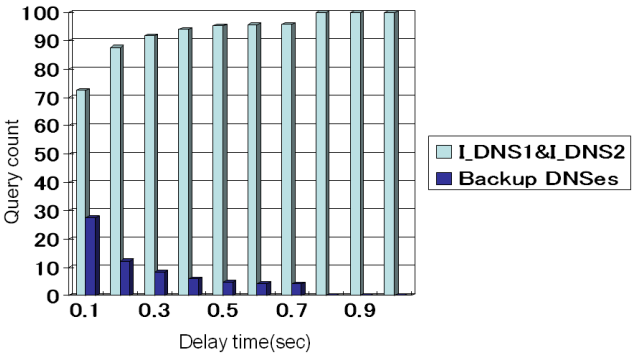

Fig. 8 DNS working status VS Delay time.

The result shows that when the delay time is not less than 0.8 second, all responses sent from the I_DNS1 and I_DNS2 (main DNS servers) become valid. In fact, we also confirmed that although all the valid DNS responses were sent from the main DNS servers, some queries were still sent to the backup servers even with not less than 0.8 second delay time set. This is because the BIND name server adds some penalty time to the lowest round trip time each time the corresponding DNS server has been chosen. Accordingly, when the round trip times of backup DNS servers become lower than those of the main DNS servers, the backup DNS servers will be chosen. In this operation, when the query is sent to the backup DNS servers initially (choosen by round robin), during waiting for response from the backup DNS servers, the client sends the same query to the main DNS server since the delay is observed on the backup DNS server. Eventually, the response from the main DNS server arrives first and becomes valid so that the response from the backup DNS server gets discarded. In other words, although the client queries the backup DNS servers periodically, the responses sent from the backup DNS servers never become valid unless a trouble occurs on the links. Furthermore, we confirmed in the preliminary experiments that the backup DNS servers received only about 5 queries in 100 time domain resolutions so that the queries to the backup DNS servers do not affect the proposal mechanism or the performance critically. Accordingly, we set 0.8 second as the delay time on the backup DNS servers in the prototype system.

Note that the 0.8 second delay time is suitable only for the experimental network environment of the prototype system and obviously we need to adjust it according to the status of a real network environment to which the proposal mechanism is being introduced, of course, under the restriction of the domain resolution timeout.

4.2 Performance Evaluation

We evaluated the performance of the proposal on the WWW and the e-mail using the network environment shown in Fig. 7. As described in Section 4.1, we set 0.8 second of delay on backup DNSes during the evaluations. For the evaluation in the WWW system, we accessed the web servers (Web1 and Web2) using the domain "www.example.com" from the External Client for 100 times with 1 second interval and confirmed the access rate of

Table 1 Evaluation results on the WWW system.

Condition	DNS RR		DRM		Proposal	
	Web1	Web2	Web1	Web2	Web1	Web2
(1) Normal status	50%	50%	50%	50%	51%	49%
Adding 2ms delay						
(2) Client→Web1	50%	50%	49%	51%	50%	50%
(3) Client→Web2	50%	50%	49%	51%	50%	50%
(4) Web1→Client	50%	50%	0%	100%	0%	100%
(5) Web2→Client	50%	50%	100%	0%	99%	1%
(6) Client↔Web1	50%	50%	0%	100%	1%	99%
(7) Client↔Web2	50%	50%	100%	0%	99%	1%
Adding 75 Mb/s FTP						
(8) Client→Web1	50%	50%	5%	95%	21%	79%
(9) Client→Web2	50%	50%	95%	5%	81%	19%
(10) Web1→Client	50%	50%	0%	100%	3%	97%
(11) Web2→Client	50%	50%	100%	0%	98%	2%

Table 2 Evaluation results on the e-mail system.

Condition	DNS RR		DRM		Proposal	
	MTA1	MTA2	MTA1	MTA2	MTA1	MTA2
(1) Normal status	50%	50%	50%	50%	49%	51%
Adding 2 ms delay						
(2) Client→MTA1	51%	49%	43%	57%	0%	100%
(3) Client→MTA2	49%	51%	58%	42%	100%	0%
(4) MTA1→Client	51%	49%	0%	100%	49%	51%
(5) MTA2→Client	49%	51%	100%	0%	49%	51%
(6) Client↔MTA1	49%	51%	0%	100%	0%	100%
(7) Client↔MTA2	51%	49%	100%	0%	100%	0%
Adding 75 Mb/s FTP						
(8) Client→MTA1	48%	52%	5%	95%	0%	100%
(9) Client→MTA2	49%	51%	95%	5%	100%	0%
(10) MTA1→Client	50%	50%	0%	100%	23%	77%
(11) MTA2→Client	51%	49%	100%	0%	76%	24%

each web server under several conditions. The conditions include a normal status (1) which means no delay or other traffics are added to the links, adding a delay in a different direction (2)–(7) and adding FTP traffic in a different direction (8)–(11). We also compared the proposal with the conventional mechanisms DNS RR and DRM to confirm its effectiveness. The evaluation results on the WWW system are shown in **Table 1**.

We can see from Table 1 that both routes were used averagely in DNS RR under all conditions. It indicates that DNS RR performed the load balancing evenly for all applications. Under the conditions (1), (2) and (3), both routes were used with nearly the same rate in the DRM and the proposal. This is because the DRM was sensitive to the outbound latency of the network for all applications but the proposal mechanism was sensitive to the outbound latency for the WWW. So that the conditions of no delay added and delay added in the inbound direction did not affect strictly the DRM and the proposal. Accordingly, we can see that under the conditions (4), (5), (6) and (7), in which the outbound delay was added, the route with no delay added was used in most times in the DRM and the proposal mechanism.

Next, when an inbound FTP traffic was added on route 1 (which had Router1) which are the conditions (8) and (9), about 95% of access in the DRM and about 80% access in the proposal were performed via the routes with no FTP traffic added. Note that the DRM is inherently sensitive to the outbound latency for all applications and the proposal mechanism was configured to be sensitive to the outbound latency for the WWW system. However, the results of conditions (8) and (9) show that both mechanisms were affected even by the inbound traffic. We consider this was because the FTP traffic (75 Mb/s) was high enough to affect both mechanisms due to some reverse direction traffic such as acknowledgment packets. Under the conditions (10) and (11), we can see that nearly all accesses were performed via the route with no FTP traffic added. This was because both of the DRM and the proposal mechanism were affected by the outbound FTP traffic sensitively. Eventually we can conclude that the system under conditions (8) and (9) is less sensitive than that under conditions (10) and (11).

We also evaluated the proposal mechanism in the e-mail system using the same network environment. In this evaluation, we sent 100 messages from the E_MTA to the Internal MGW (MTA1 and MTA2) using the domain name "mail.example.com" with 1 second interval and confirmed the utilization rate of each MGW under several conditions. Likewise, we also compared the proposal mechanism with the conventional mechanisms DNS RR and DRM to confirm its effectiveness. The evaluation results on the e-mail system are shown in **Table 2**.

We can see from Table 2 that both routes were used nearly with the same rate in DNS RR under all conditions as always. Under the conditions (4) and (5), we can see that all e-mails were delivered via the route with no delay added in the DRM while both routes were used almost evenly in the proposal mechanism. This was because the DRM is sensitive to the outbound latency for all applications while the proposal mechanism is sensitive to the inbound latency for inbound e-mail delivery in the e-mail system. Under the conditions (2) and (3), we can see that both routes were used with nearly the same rate in the DRM while most e-mails were delivered via the route with no delay added in the proposal mechanism. This also indicates that the proposal mechanism worked well based on the network condition in the proper direction according to application characteristics. Under the conditions (6) and (7), we can see that most e-mails were delivered via the route with no delay added in the DRM and the proposal mechanism. This was because the delay was added in a bi-directional manner so that both methods were affected by the delay time.

Next, when an inbound FTP traffic was added on route 1 (which has Router1) which are the conditions (8) and (9), we can see that most e-mails were delivered via the route with no FTP traffic added in the DRM and all e-mails were delivered via the route with no FTP traffic added in the proposal mechanism. This was because the proposal was affected by the inbound FTP traffic. On the other hand, although the DRM was sensitive to the outbound latency for all applications but the inbound FTP traffic (75 Mb/s) was high enough to affect the DRM. Under the conditions (10) and (11), we can see that all e-mails were delivered via the route with no FTP traffic added in the DRM while only about 75% e-mails were delivered via the route with no FTP traffic added in the proposal mechanism. This indicates again that the proposal mechanism worked properly based on application

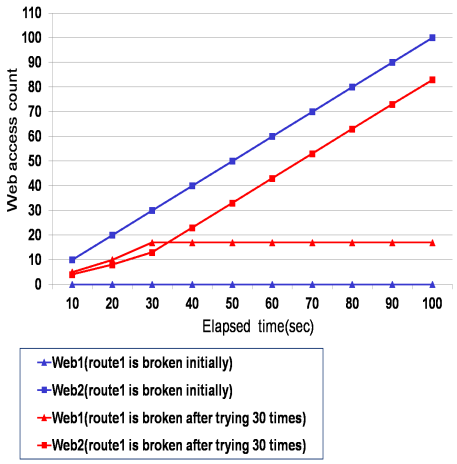

Fig. 9 Results for fault-tolerance feature on the WWW system.

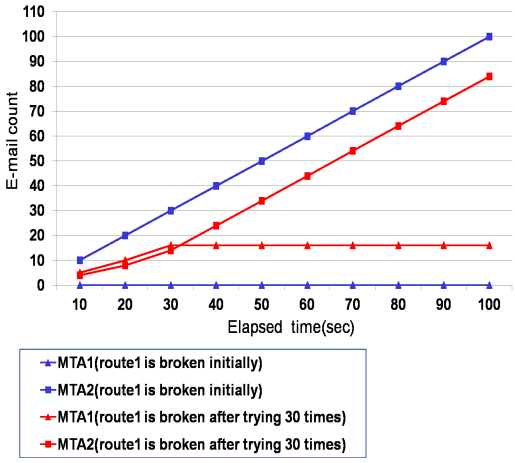

Fig. 10 Results for fault-tolerance feature on the e-mail system.

characteristics and was sensitive to the inbound latency for the inbound delivery in the e-mail system while the DRM was sensitive to the outbound latency for all applications.

4.3 Evaluation of Fault-tolerance

More importantly, we also evaluated the fault-tolerance feature on the e-mail system and the WWW system using the same network configuration. We sent 100 e-mails from the external mail server to the internal mail servers (See Fig. 6) with one second interval and verified the selected routes through which the e-mails were delivered under two situations: one of the routes was broken initially and one of the routes was broken after 30 seconds had passed. The evaluation result is shown in **Fig. 9** and **Fig. 10**. In Fig. 9, the X-axis means the elapsed time and the Y-axis means the count of web accesses. Similarly in Fig. 10, the X-axis describes the elapsed time and the Y-axis describes the count of e-mails. As we can see from the results, firstly, when one of the routes was broken initially, all e-mails were delivered via the other route and all WWW accesses were performed via the other route. Secondly, when one of the routes was broken in the middle of service, the rest of e-mails were delivered via the other route and none of them were discarded. Similarly, in the WWW system, the rest of accesses were performed via the other route and none of them was failed. Consequently, we confirm that the fault-tolerance feature of the proposal mechanism worked well on a multihomed network as expected.

Table 3 The overhead of domain name resolution.

Initial try of the normal case (Not cached in E_DNS)	506 ms
Second try and after in the normal case (Cached in E_DNS)	2 ms
Initial try of the proposal (See Fig. 5)	1,016 ms
Second try and after of the proposal	10 ms

According to the above evaluation results we confirm that the proposal mechanism worked well on both the inbound type and the outbound type applications and was more effective than conventional route selection mechanisms. Moreover, even in the network environment with ingress filtering, the proposal mechanism could still work well while the DRM would still reveal problems on it.

4.4 Overhead

Finally, we also measured the overhead of the domain name resolution in a normal DNS system and the proposal mechanism using the same experimental network. The results are shown in **Table 3**. At the initial try in the normal case, that is, when the proposal mechanism was not applied and the zone information was not cached in the E_DNS, it took about 506 ms. In other words, in this procedure the E_DNS did not have any zone information about the target domain name before the domain resolution. Thus, this procedure included the following four steps: the client queried the E_DNS, the E_DNS queried the RootDNS, the E_DNS queried an internal DNS server of the target domain and finally the E_DNS replied with the DNS response to the client. However, after the initial try of domain name resolution, the E_DNS cached the zone information of the target domain so that it took about 2 ms on the second try and after. On the other hand, in the proposal mechanism, it took about 1,016 ms at the initial try, including all the 10 steps shown in Fig. 5. On the second try and after, it took about 10 ms since the zone information was cached in the E_DNS. According to the above results, although the overhead in the proposal mechanism was a little bit high, it is acceptable for practical use and possibly able to be improved by a different implementation technology.

5. Conclusion

In this paper, we discussed the conventional route selection mechanisms on multihomed networks as well as their problems and proposed a new adaptive route selection mechanism based on the multipath DNS round trip time to solve the existing problems in conventional mechanisms. We also implemented the prototype system for the proposal mechanism and evaluated the features on the WWW and e-mail systems. According to the evaluation results, we confirm that the prototype system worked effectively and solved the existing problems.

The future works include the performance evaluation in real network environments as well as a review on other Internet services such as FTP.

Reference

[1] Yamai, N., Okayama, K., Shimamoto, H. and Okamoto, T.: A dynamic traffic sharing with minimal administration on multihomed networks, *Proc. IEEE International Conference on Communications 2001* (*ICC2001*), Vol.5, pp.1506–1510, Helsinki, Finland, IEEE-

ComSoc (2001).
[2] Hawkinson, J. and Bates, T.: Guidelines for creation, selection, and registration of an Autonomous System (AS), RFC 1930, IETF (1996).
[3] Yamai, N., Doi, M., Okayama, K. and Nakamura, M.: A Reliable Operation Method of E-mail Systems on Multihomed Networks (in Japanese), *Information Technology Letters*, Vol.6, pp.373–376, Toyota, Japan, IPSJ/IEICE-ISS/IEICE-HCG (2007).
[4] Brisco, T.: DNS Support for Load Balancing, RFC 1794, IETF (1995).
[5] Jin, Y., Yamai, N., Okayama, K. and Nakamura, M.: A dynamic route selection mechanism using multiple DNS responses for inbound e-mail delivery on multihomed networks, *Proc. International Conference on Information Networking 2010* (*ICOIN 2010*), No.6A-4, Pusan, Korea, KIISE (2010).
[6] Baker, F. and Savola, P.: Ingress Filtering for Multihomed Networks, RFC 3704, IETF (2004).
[7] Rekhter, Y., Li, T. and Hares, S. (Eds.): A Border Gateway Protocol 4 (BGP-4), RFC 4271, IETF (2006).
[8] Internet Systems Consortium, Inc.: BIND — Internet Systems Consortium (online), available from ⟨https://www.isc.org/software/bind⟩ (accessed 2011-06-23).
[9] Delgadillo, K.: Cisco DistributedDirector, Cisco Systems, Inc. (online), available from ⟨http://www.cisco.com/warp/public/cc/pd/cxsr/dd/tech/dd_wp.pdf⟩ (accessed 2011-06-23).
[10] Shimokawa, T., Koba, Y., Nakagawa, I., Yamamoto, B. and Yoshida, N.: Server Selection Mechanism using DNS and Routing Information in Widely Distributed Environment (in Japanese), *IEICE Trans. Comm.*, Vol.J86-B, No.8, pp.1454–1462 (2003).
[11] Mills, D., Martin, J., Burbank, J. and Kasch, W.: Network Time Protocol Version 4: Protocol and Algorithms Specification, RFC 5905, IETF (2010).
[12] Brown, R.: Net::DNSServer::Proxy, Perl.org (online), available from ⟨http://search.cpan.org/~bbb/Net-DNSServer-0.11/lib/Net/DNSServer/Proxy.pm⟩ (accessed 2011-06-23).
[13] Rizzo, L.: dummynet - traffic shaper, bandwidth manager and delay emulator, FreeBSD Kernel Interface Manual (online), available from ⟨http://www.freebsd.org/cgi/man.cgi?query=dummynet⟩ (accessed 2011-09-26).

Yong Jin received his M.E. degree in electronic and information systems engineering from Okayama Univeristy, Japan, in 2009. Since April 2009, he has been a Ph.D. candidate in industrial innovation sciences of Okayama University. His research intreastes include Distributed System, Network Architecture and Internet.

Nariyoshi Yamai received his B.E. and M.E. degrees in electronic engineering and his Ph.D. degree in information and computer science from Osaka University, Osaka, Japan, in 1984, 1986 and 1993, respectively. In April 1988, he joined the Department of Information Engineering, Nara National College of Technology, as a research associate. From April 1990 to March 1994, he was an assistant professor in the same department. In April 1994, he joined the Education Center for Information Processing, Osaka University, as a Research Associate. In April 1995, he joined the Computation Center, Osaka University, as an assistant professor. From November 1997 to March 2006, he joined the Computer Center, Okayama University, as an associate professor. Since April 2006, he has been a professor in Information Technology Center (at present, Center for Information Technology and Management), Okayama University. His research interests include Distributed System, Network Security, and Internet. He is a member of IEICE and IEEE.

Kiyohiko Okayama received his B.S., M.S. and Ph.D. degrees in information and computer sciences from Osaka University, Japan, in 1990, 1992 and 2001, respectively. After he has worked in the Department of Information System at Osaka University and in the Graduate School of Information Science at Nara Institute of Science and Technology as a research associate, he joined the Department of Communication Network Engineering at Okayama University in 2000. From 2005 to 2011, he joined the Information Technology Center at Okayama University. Since 2011, he has been an associate professor in Center for Information Technology and Management at Okayama University. His research interests include network design and network security. He is a member of IEICE.

Motonori Nakamura graduated from Kyoto University, Japan, where he received his B.E., M.E. and Ph.D. degrees in engineering in 1989, 1991 and 1996, respectively. From 1994, he was an assistant professor at Ritsumeikan University. From 1995, he was an associate professor at Kyoto University. Currently he is a professor at National Institute of Informatics, Japan (NII) and the Graduate University for Advanced Studies (SOKENDAI). His research interests are message transport network systems, network communications, next generation internet and Identity & Access Management. He is a member of IEEE, ISOC, IEICE and JSSST.